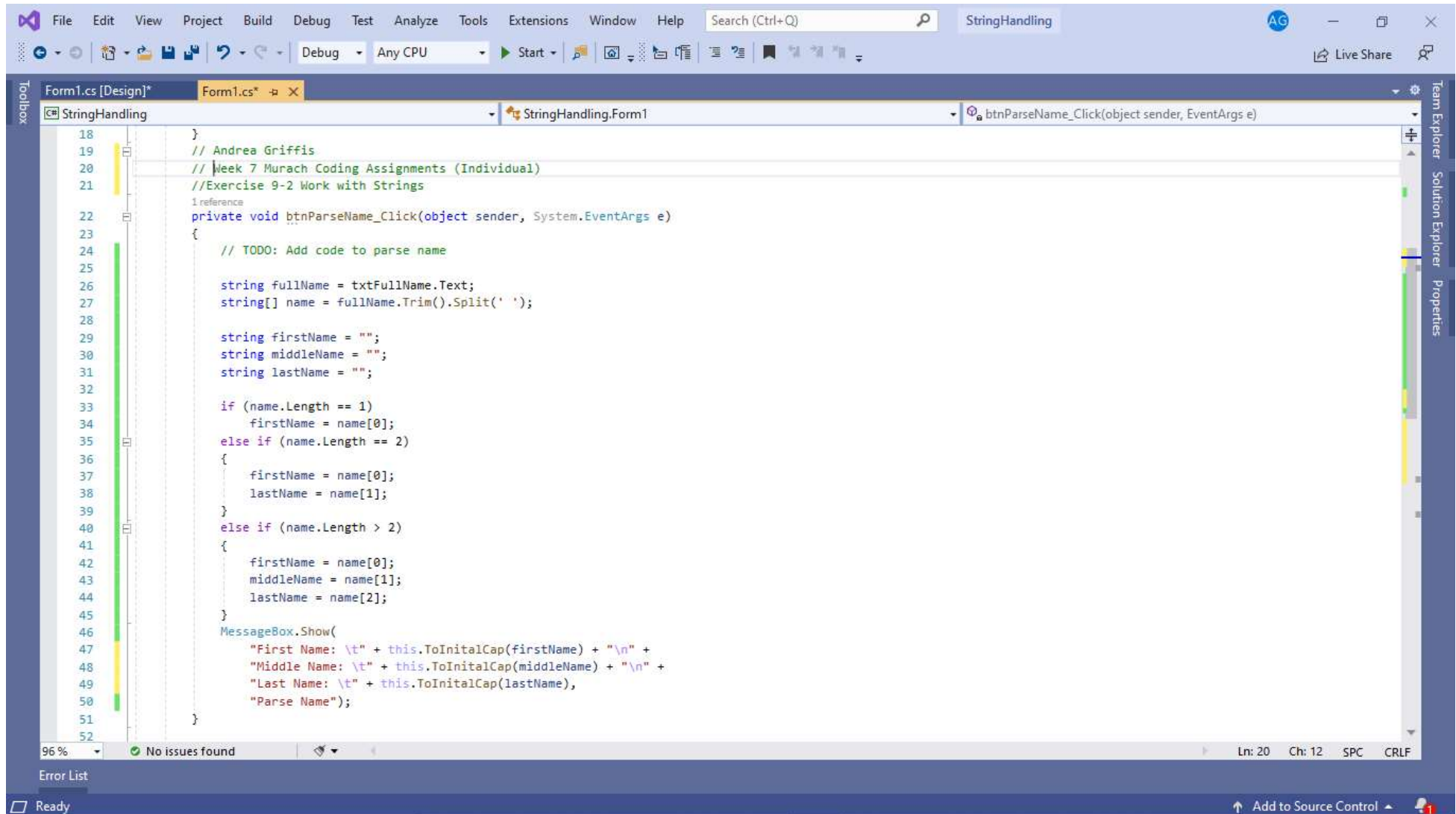


Assignment: Week 7 Murach Coding Assignments (Individual)

Exercise 9-2 Work with Strings

Add code to parse a name



The screenshot shows the Visual Studio IDE with a C# project named 'StringHandling'. The file 'Form1.cs' is open, and the 'StringHandling.Form1' class is selected. The 'btnParseName_Click' event handler is being edited. The code implements a function to parse a full name into first, middle, and last names based on the number of spaces. It uses 'Trim()' and 'Split(' ')" to process the input string. A 'MessageBox.Show()' call displays the parsed components with their first letters capitalized.

```
18 }
19 // Andrea Griffis
20 // Week 7 Murach Coding Assignments (Individual)
21 // Exercise 9-2 Work with Strings
22 1 reference
23 private void btnParseName_Click(object sender, EventArgs e)
24 {
25     // TODO: Add code to parse name
26
27     string fullName = txtFullName.Text;
28     string[] name = fullName.Trim().Split(' ');
29
30     string firstName = "";
31     string middleName = "";
32     string lastName = "";
33
34     if (name.Length == 1)
35         firstName = name[0];
36     else if (name.Length == 2)
37     {
38         firstName = name[0];
39         lastName = name[1];
40     }
41     else if (name.Length > 2)
42     {
43         firstName = name[0];
44         middleName = name[1];
45         lastName = name[2];
46     }
47     MessageBox.Show(
48         "First Name: \t" + this.ToInitialCap(firstName) + "\n" +
49         "Middle Name: \t" + this.ToInitialCap(middleName) + "\n" +
50         "Last Name: \t" + this.ToInitialCap(lastName),
51         "Parse Name");
52 }
```

96 % No issues found Ln: 20 Ch: 12 SPC CRLF

Ready Add to Source Control

Visual Studio interface showing the code for `StringHandling` in `Form1.cs`. The code implements a `btnParseName_Click` event handler and a `ToInitialCap` method.

```
29 string firstName = "";
30 string middleName = "";
31 string lastName = "";
32
33 if (name.Length == 1)
34     firstName = name[0];
35 else if (name.Length == 2)
36 {
37     firstName = name[0];
38     lastName = name[1];
39 }
40 else if (name.Length > 2)
41 {
42     firstName = name[0];
43     middleName = name[1];
44     lastName = name[2];
45 }
46
47 MessageBox.Show(
48     "First Name: \t" + this.ToInitialCap(firstName) + "\n" +
49     "Middle Name: \t" + this.ToInitialCap(middleName) + "\n" +
50     "Last Name: \t" + this.ToInitialCap(lastName),
51     "Parse Name");
52
53 private string ToInitialCap(string upper)
54 {
55     if (upper.Length > 0)
56     {
57         string initialCap = upper.Substring(0, 1).ToUpper();
58         string lwrcaseLetters = upper.Substring(1).ToLower();
59         upper = initialCap + lwrcaseLetters;
60     }
61     return upper;
62 }
63 }
```

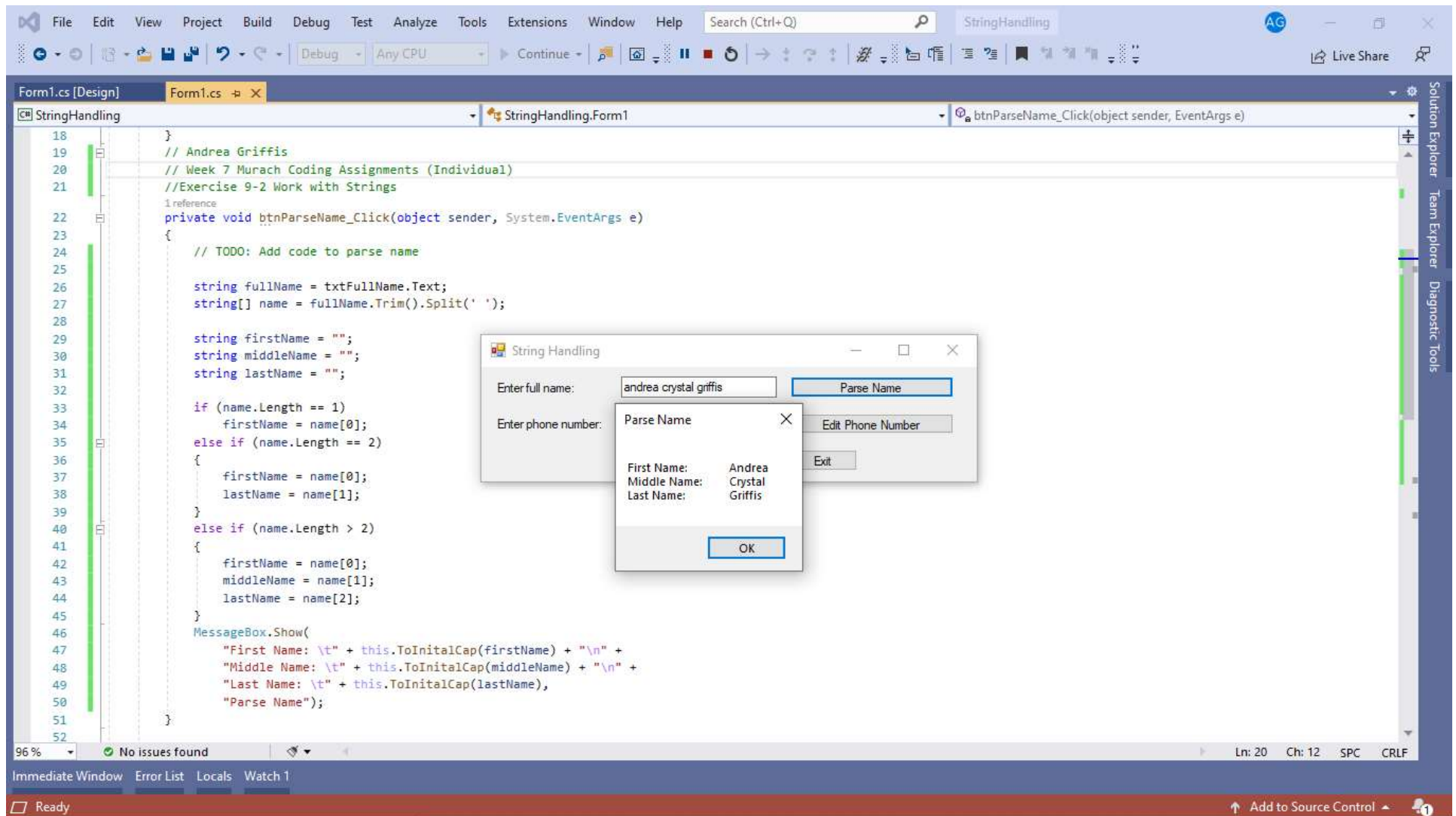
3 references

96 % No issues found Ln: 20 Ch: 12 SPC CRLF

Immediate Window Error List Locals Watch 1

Ready Add to Source Control

Test the Application



Visual Studio interface showing a C# project named **StringHandling**. The main window displays the code for **Form1.cs** in Design mode, with the **StringHandling.Form1** class selected. The code implements a **btnParseName_Click** event handler that parses a full name into first, middle, and last names and displays the results in a message box.

```
18 }
19 // Andrea Griffis
20 // Week 7 Murach Coding Assignments (Individual)
21 //Exercise 9-2 Work with Strings
22 1 reference
23 private void btnParseName_Click(object sender, EventArgs e)
24 {
25     // TODO: Add code to parse name
26
27     string fullName = txtFullName.Text;
28     string[] name = fullName.Trim().Split(' ');
29
30     string firstName = "";
31     string middleName = "";
32     string lastName = "";
33
34     if (name.Length == 1)
35         firstName = name[0];
36     else if (name.Length == 2)
37     {
38         firstName = name[0];
39         lastName = name[1];
40     }
41     else if (name.Length > 2)
42     {
43         firstName = name[0];
44         middleName = name[1];
45         lastName = name[2];
46     }
47     MessageBox.Show(
48         "First Name: \t" + this.ToInitialCap(firstName) + "\n" +
49         "Middle Name: \t" + this.ToInitialCap(middleName) + "\n" +
50         "Last Name: \t" + this.ToInitialCap(lastName),
51         "Parse Name");
52 }
```

The **String Handling** dialog box is open, showing the input "ANDREA CRYSTAL GRIFFIS" and the "Parse Name" button. The **Parse Name** dialog box is also open, displaying the parsed names: First Name: Andrea, Middle Name: Crystal, Last Name: Griffis, and the "OK" button.

The status bar at the bottom indicates "96 %", "No issues found", and "Ln: 20 Ch: 12 SPC CRLF". The bottom-most bar shows "Ready" and "Add to Source Control".

Add code to edit a phone number

The screenshot shows the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q)
- Toolbox:** StringHandling
- Form Explorer:** StringHandling.Form1
- Code Editor:** Form1.cs [Design]*, Form1.cs*
 - Line 51: `}`
 - Line 52: `// Andrea Griffis`
 - Line 53: `// Week 7 Murach Coding Assignments (Individual)`
 - Line 54: `//Exercise 9-2 Work with Strings`
 - Line 55: `1 reference`
 - Line 56: `private void btnEditPhoneNumber_Click(object sender, EventArgs e)`
 - Line 57: `{`
 - Line 58: `// TODO: Add code to edit the phone number`
 - Line 59: `string phoneNum = txtPhoneNumber.Text.Trim();`
 - Line 60: `string numbers = "";`
 - Line 61: `foreach (char c in phoneNum)`
 - Line 62: `{`
 - Line 63: `if (!(`
 - Line 64: `c == '(' || c == ')' ||`
 - Line 65: `c == '-' || c == '.' ||`
 - Line 66: `c == ' ' || c == '-' || c == '.'`
 - Line 67: `))`
 - Line 68: `{`
 - Line 69: `numbers += c;`
 - Line 70: `}`
 - Line 71: `}`
 - Line 72: `string standardFmt = numbers.Insert(3, "-");`
 - Line 73: `standardFmt = standardFmt.Insert(7, "-");`
 - Line 74: `MessageBox.Show(`
 - Line 75: `"Entered: \t\t" + phoneNum + "\n\n" +`
 - Line 76: `"Digits Only: \t\t" + numbers + "\n\n" +`
 - Line 77: `"Standard Format: \t\t" + standardFmt,`
 - Line 78: `"Edit Phone Number");`
 - Line 79: `}`
 - Line 80: `// TODO: Add ToInitialCap method here`
 - Line 81: `3 references`
 - Line 82: `private string ToInitialCap(string upper)`
 - Line 83: `{`
 - Line 84: `if (upper.Length > 0)`
 - Line 85: `{`

The status bar at the bottom shows: 96 %, No issues found, Ln: 54, Ch: 41, SPC, CRLF.

Test the application

The screenshot shows the Visual Studio IDE with the `StringHandling` project open. The `Form1.cs` file is in Design view, showing the `btnEditPhoneNumber_Click` event handler. The code processes a phone number by inserting dashes. Two dialog boxes are displayed: 'String Handling' for input and 'Edit Phone Number' for output.

```
51 }
52 // Andrea Griffis
53 // Week 7 Murach Coding Assignments (Individual)
54 //Exercise 9-2 Work with Strings
55 1 reference
56 private void btnEditPhoneNumber_Click(object sender, EventArgs e)
57 {
58     // TODO: Add code to edit the phone number
59     string phoneNum = txtPhoneNumber.Text.Trim();
60     string numbers = "";
61     foreach (char c in phoneNum)
62     {
63         if (!(
64             c == '(' || c == ')' ||
65             c == '-' || c == '.' || c == ' ')
66         {
67             numbers += c;
68         }
69     }
70
71     string standardFmt = numbers.Insert(3, "-");
72     standardFmt = standardFmt.Insert(7, "-");
73     MessageBox.Show(
74         "Entered: \t\t" + phoneNum + "\n\n" +
75         "Digits Only: \t\t" + numbers + "\n\n" +
76         "Standard Format: \t\t" + standardFmt,
77         "Edit Phone Number");
78 }
79
80 // TODO: Add ToInitialCap method here
81 3 references
82 private string ToInitialCap(string upper)
83 {
84     if (upper.Length > 0)
85     {
```

String Handling

Enter full name: joel ray murach Parse Name

Enter phone number: (559) 440-9071 Edit Phone Number

Edit Phone Number

Entered: (559) 440-9071

Digits Only: 5594409071

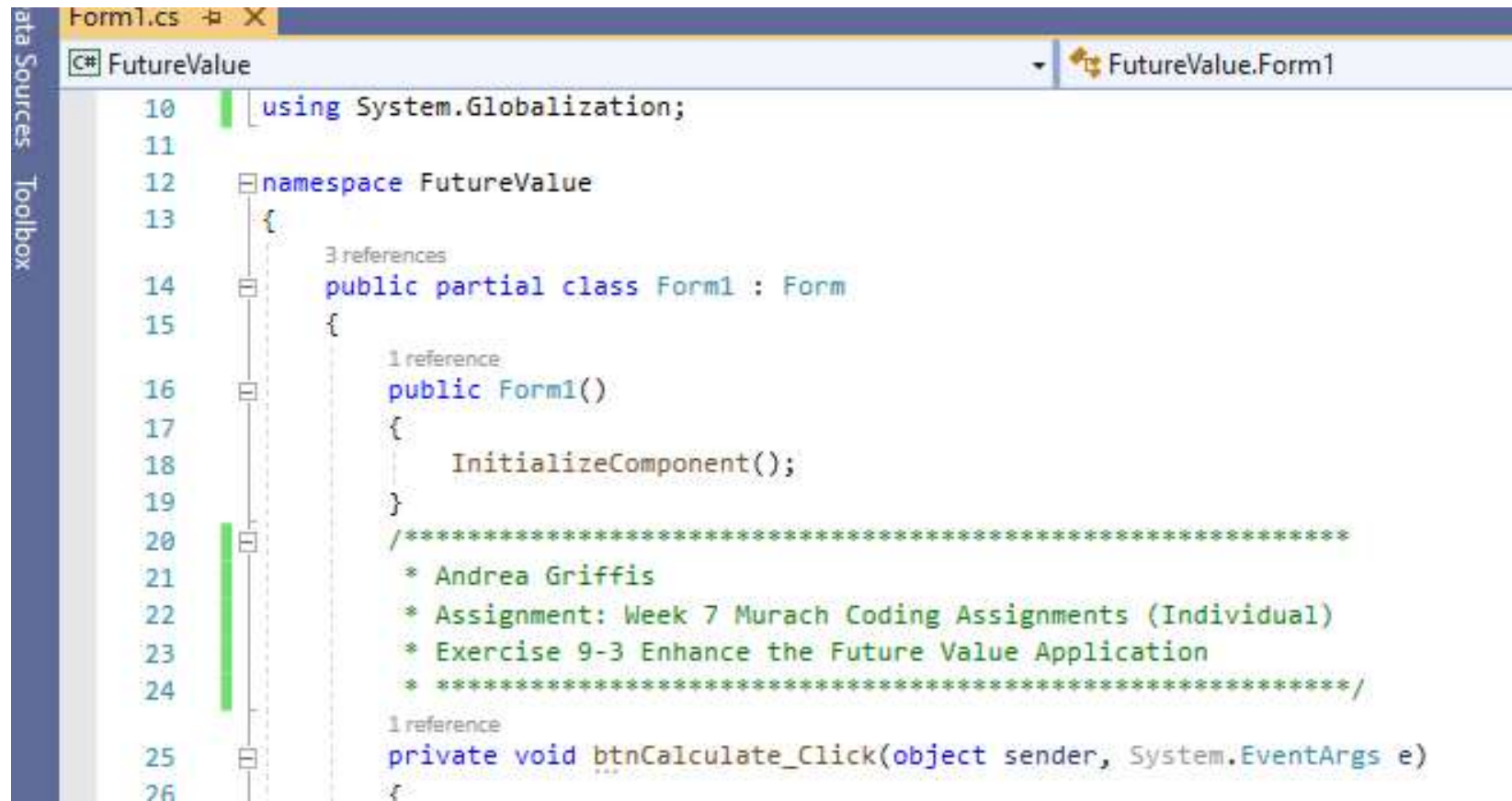
Standard Format: 559-440-9071

OK

Exercise 9-3 Enhance the Future Value application

Add code that provides for formatted enteries (#'s 2- 10)

#2



The screenshot shows the Visual Studio IDE with the 'FutureValue' project selected. The 'Form1.cs' file is open, displaying the following code:

```
10 using System.Globalization;
11
12 namespace FutureValue
13 {
14     public partial class Form1 : Form
15     {
16         public Form1()
17         {
18             InitializeComponent();
19         }
20         /*****
21          * Andrea Griffis
22          * Assignment: Week 7 Murach Coding Assignments (Individual)
23          * Exercise 9-3 Enhance the Future Value Application
24          * *****/
25         private void btnCalculate_Click(object sender, EventArgs e)
26         {
```

The code is written in C# and defines a partial class `Form1` within the `FutureValue` namespace. It includes a constructor `Form1()` that calls `InitializeComponent()`. A comment block is present between lines 20 and 24, identifying the author as Andrea Griffis and the assignment as 'Exercise 9-3 Enhance the Future Value Application'. The `btnCalculate_Click` event handler is also visible, starting at line 25.

#3 and #4

Visual Studio interface showing the code for `FutureValue.Form1`. The code includes comments about the assignment and two methods: `IsDecimal` and `IsCurrency`.

```
86 * Andrea Griffis
87 * Assignment: Week 7 Murach Coding Assignments (Individual)
88 * Exercise 9-3 Enhance the Future Value Application
89 * *****/
90 // TODO: improve this IsDecimal method
91 2 references
92 public bool IsDecimal(TextBox textBox, string name)
93 {
94     decimal number = 0m;
95     if (Decimal.TryParse(textBox.Text, NumberStyles.Number, CultureInfo.CurrentCulture, out number))
96     {
97         return true;
98     }
99     else
100     {
101         MessageBox.Show(name + " must be a decimal value.", "Entry Error");
102         textBox.Focus();
103         return false;
104     }
105 }
106 // TODO: add a new method to test for currency entries
107 0 references
108 public bool IsCurrency(TextBox textBox, string name)
109 {
110     decimal number = 0m;
111     if (Decimal.TryParse(textBox.Text, NumberStyles.Currency, CultureInfo.CurrentCulture, out number))
112     {
113         return true;
114     }
115     else
116     {
117         MessageBox.Show(name + " must be in currency format.", "Enter Error");
118         textBox.Focus();
119         return false;
120     }
121 }
```

Status bar: 96 % No issues found Ln: 104 Ch: 10 SPC CRLF

Item(s) Saved

#5 and #6

Visual Studio interface showing the code for `FutureValue.Form1`. The code includes comments about the assignment and two methods: `IsInt32` and `IsWithinRange`.

```
117 * Andrea Griffis
118 * Assignment: Week 7 Murach Coding Assignments (Individual)
119 * Exercise 9-3 Enhance the Future Value Application
120 * *****/
121
122 // TODO: improve this IsInt32 method
123 1 reference
124 public bool IsInt32(TextBox textBox, string name)
125 {
126     int number = 0;
127     if (Int32.TryParse(textBox.Text, NumberStyles.None, CultureInfo.CurrentCulture, out number))
128     {
129         return true;
130     }
131     else
132     {
133         MessageBox.Show(name + " must be an integer.", "Entry Error");
134         textBox.Focus();
135         return false;
136     }
137 }
138 // Updated to Parse method to allow all numbers and styling
139 3 references
140 public bool IsWithinRange(TextBox textBox, string name,
141     decimal min, decimal max)
142 {
143     decimal number = Decimal.Parse(textBox.Text, NumberStyles.Currency);
144     if (number < min || number > max)
145     {
146         MessageBox.Show(name + " must be between " + min +
147             " and " + max + ".", "Entry Error");
148         textBox.Focus();
149         return false;
150     }
151     return true;
152 }
```

96 % No issues found Ln: 121 Ch: 9 SPC CRLF

Error List

Ready Add to Source Control

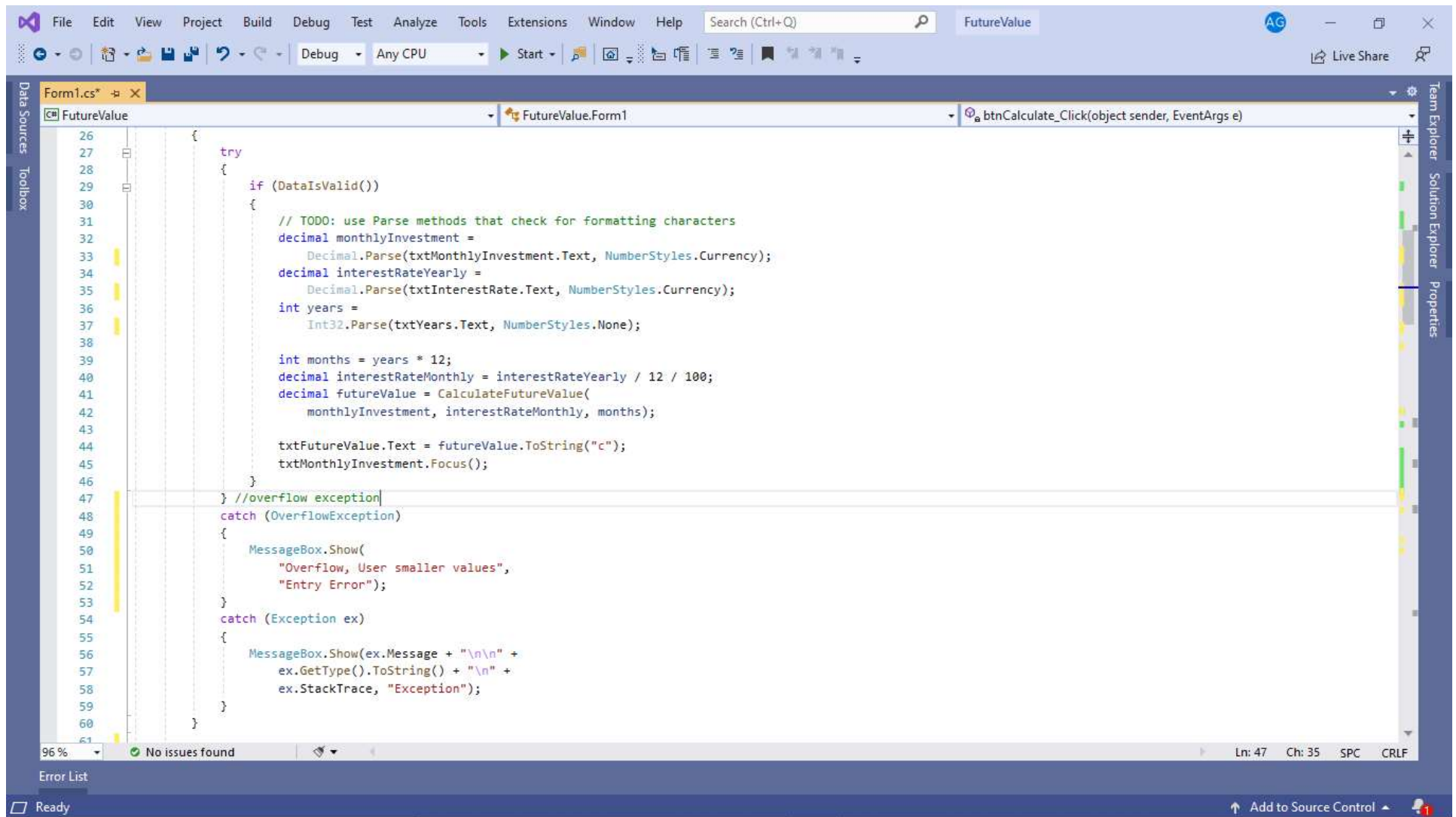
#7

The screenshot shows the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Project, Build, Debug, Test, Analyze, Tools, Futu...alue, Extensions, Window, Help.
- Toolbar:** Includes icons for file operations, a search icon, and a Live Share button.
- Left Sidebar:** Contains 'Data Sources' and 'Toolbox' tabs.
- Right Sidebar:** Contains 'Team Explorer', 'Solution Explorer', and 'Properties' tabs.
- Code Editor:** Displays the code for 'Form1.cs' in the 'FutureValue' namespace, specifically the 'DatsValid()' method of 'FutureValue.Form1'. The code includes comments about the assignment and validation logic for monthly investment, interest rate, and number of years.
- Bottom Bar:** Shows '96 %' zoom, 'No issues found', and a status bar with 'Ln: 60 Ch: 8 SPC CRLF'.
- Bottom Panel:** Labeled 'Error List', it is currently empty.
- Footer:** Includes an 'Add to Source Control' button and a notification icon.

```
52         ex.StackTrace, "Exception");
53     }
54 }
55
56 /*****
57  * Andrea Griffis
58  * Assignment: Week 7 Murach Coding Assignments (Individual)
59  * Exercise 9-3 Enhance the Future Value Application
60  * *****/
61
62 // Updated IsDecimal to IsCurrency for the Monthly Investment
63 1 reference
64 public bool DataIsValid()
65 {
66     return
67         // Validate the Monthly Investment text box
68         IsPresent(txtMonthlyInvestment, "Monthly Investment") &&
69         IsCurrency(txtMonthlyInvestment, "Monthly Investment") &&
70         IsWithinRange(txtMonthlyInvestment, "Monthly Investment", 1, 1000) &&
71
72         // Validate the Yearly Interest Rate text box
73         IsPresent(txtInterestRate, "Yearly Interest Rate") &&
74         IsDecimal(txtInterestRate, "Yearly Interest Rate") &&
75         IsWithinRange(txtInterestRate, "Yearly Interest Rate", 1, 20) &&
76
77         // Validate the Number of Years text box
78         IsPresent(txtYears, "Number of Years") &&
79         IsInt32(txtYears, "Number of Years") &&
80         IsWithinRange(txtYears, "Number of Years", 1, 40);
81 }
```

#8 and 9



The screenshot shows the Visual Studio IDE with the 'FutureValue' project open. The code editor displays the 'btnCalculate_Click' method in 'Form1.cs'. The code is as follows:

```
26 {
27     try
28     {
29         if (DataIsValid())
30         {
31             // TODO: use Parse methods that check for formatting characters
32             decimal monthlyInvestment =
33                 Decimal.Parse(txtMonthlyInvestment.Text, NumberStyles.Currency);
34             decimal interestRateYearly =
35                 Decimal.Parse(txtInterestRate.Text, NumberStyles.Currency);
36             int years =
37                 Int32.Parse(txtYears.Text, NumberStyles.None);
38
39             int months = years * 12;
40             decimal interestRateMonthly = interestRateYearly / 12 / 100;
41             decimal futureValue = CalculateFutureValue(
42                 monthlyInvestment, interestRateMonthly, months);
43
44             txtFutureValue.Text = futureValue.ToString("c");
45             txtMonthlyInvestment.Focus();
46         }
47     } //overflow exception
48     catch (OverflowException)
49     {
50         MessageBox.Show(
51             "Overflow, User smaller values",
52             "Entry Error");
53     }
54     catch (Exception ex)
55     {
56         MessageBox.Show(ex.Message + "\n\n" +
57             ex.GetType().ToString() + "\n\n" +
58             ex.StackTrace, "Exception");
59     }
60 }
61 }
```

The status bar at the bottom indicates '96 %', 'No issues found', and 'Ln: 47 Ch: 35 SPC CRLF'. The 'Error List' pane is empty, and the 'Ready' status is shown.

#10 Test the application

```

    double monthlyInvestment = double.Parse(txtMonthlyInvestment.Text, NumberStyles.Currency);
    double interestRateYearly = double.Parse(txtInterestRate.Text, NumberStyles.Currency);
    double years = double.Parse(txtYears.Text, NumberStyles.Currency);

    double years * 12;
    double interestRateMonthly = interestRateYearly / 12;
    double futureValue = CalculateFutureValue(monthlyInvestment, interestRateMonthly, years);

    txtFutureValue.Text = futureValue.ToString("C");
    txtFutureValue.Focus();
}

```

Future Value

Monthly Investment:

Yearly Interest Rate:

Number of Years:

Future Value:

Enter Error

Monthly Investment must be in currency format.

```

    }
}

```

```

    "smaller values",
    ;
}

```

```

    double interestRateYearly = double.Parse(txtInterestRate.Text, NumberStyles.Currency);
    double years = double.Parse(txtYears.Text, NumberStyles.Currency);

    double years * 12;
    double interestRateMonthly = interestRateYearly / 12;
    double futureValue = CalculateFutureValue(monthlyInvestment, interestRateMonthly, years);

    txtFutureValue.Text = futureValue.ToString("C");
    txtFutureValue.Focus();
}

```

Future Value

Monthly Investment:

Yearly Interest Rate:

Number of Years:

Future Value:

Enter Error

Monthly Investment must be in currency format.

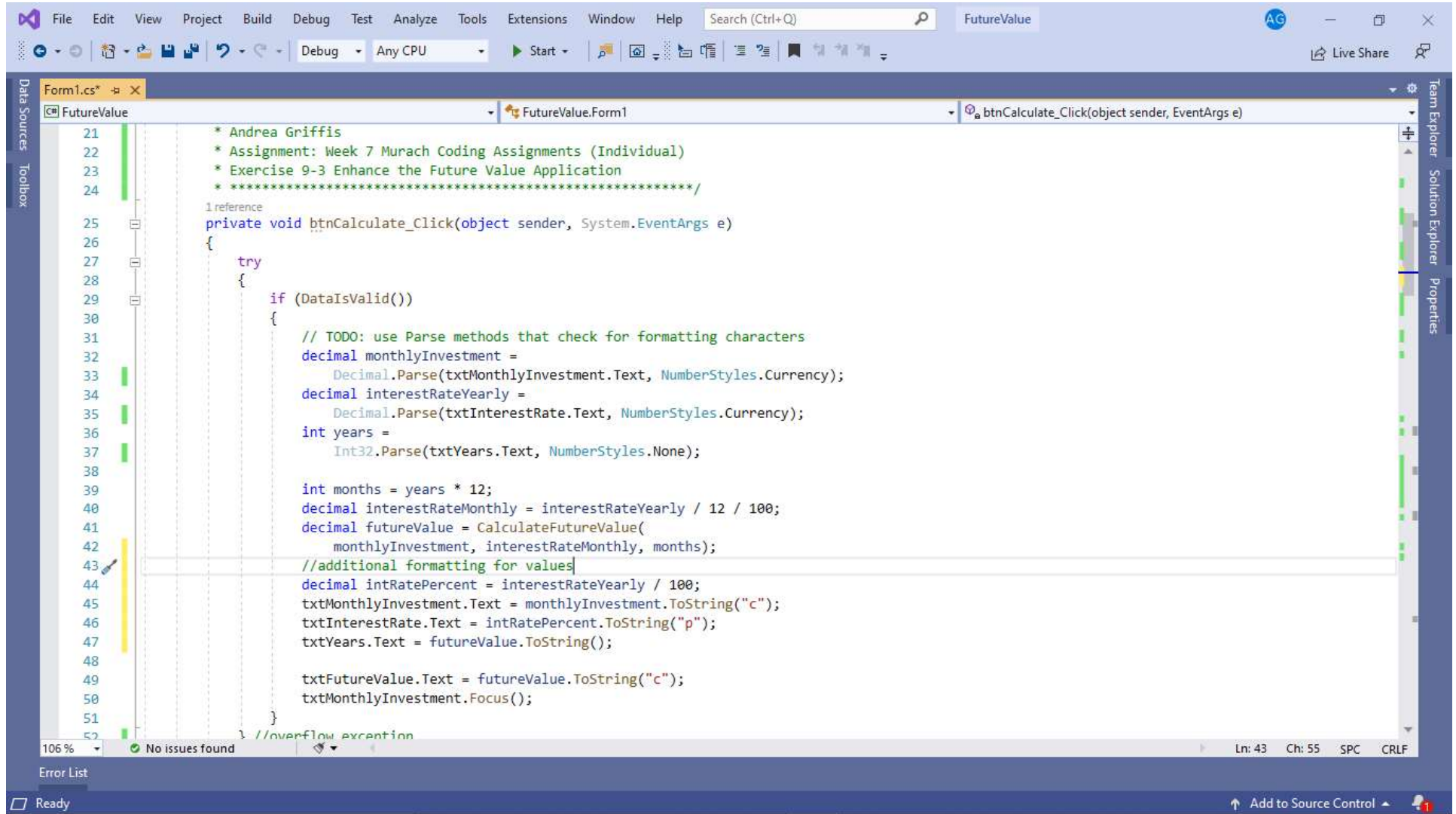
```

    "smaller values",
    ;
}

```


Add code to format the displayed values

#11



The screenshot shows the Visual Studio IDE with a C# file named `Form1.cs` open. The code is for a Windows Form application titled `FutureValue`. The `btnCalculate_Click` event handler is currently selected in the Solution Explorer. The code calculates the future value based on monthly investment, interest rate, and years, and then formats the results for display in text boxes.

```
21 * Andrea Griffis
22 * Assignment: Week 7 Murach Coding Assignments (Individual)
23 * Exercise 9-3 Enhance the Future Value Application
24 * *****/
25 1 reference
26 private void btnCalculate_Click(object sender, EventArgs e)
27 {
28     try
29     {
30         if (DataIsValid())
31         {
32             // TODO: use Parse methods that check for formatting characters
33             decimal monthlyInvestment =
34                 Decimal.Parse(txtMonthlyInvestment.Text, NumberStyles.Currency);
35             decimal interestRateYearly =
36                 Decimal.Parse(txtInterestRate.Text, NumberStyles.Currency);
37             int years =
38                 Int32.Parse(txtYears.Text, NumberStyles.None);
39
40             int months = years * 12;
41             decimal interestRateMonthly = interestRateYearly / 12 / 100;
42             decimal futureValue = CalculateFutureValue(
43                 monthlyInvestment, interestRateMonthly, months);
44             //additional formatting for values
45             decimal intrRatePercent = interestRateYearly / 100;
46             txtMonthlyInvestment.Text = monthlyInvestment.ToString("c");
47             txtInterestRate.Text = intrRatePercent.ToString("p");
48             txtYears.Text = futureValue.ToString();
49
50             txtFutureValue.Text = futureValue.ToString("c");
51             txtMonthlyInvestment.Focus();
52         }
53     }
54     //overflow exception
55 }
```

The status bar at the bottom indicates "106 %", "No issues found", and "Ln: 43 Ch: 55 SPC CRLF".

Test the application

When entered with only numbers

Future Value

Monthly Investment:

\$100.00

Yearly Interest Rate:

4.00%

Number of Years:

10

Future Value:

\$14,774.06

Calculate

Exit

When entered with % sign in interest rate and monthly investment

Future Value

Monthly Investment:

\$100.00

Yearly Interest Rate:

4.00%

Number of Years:

10

Future Value:

\$14,774.06

Calculate

Exit

Entry Error

Yearly Interest Rate must be a decimal value.

OK

Future Value

Monthly Investment:

100%

Yearly Interest Rate:

4

Number of Years:

10

Future Value:

\$14,774.06

Calculate

Exit

Enter Error

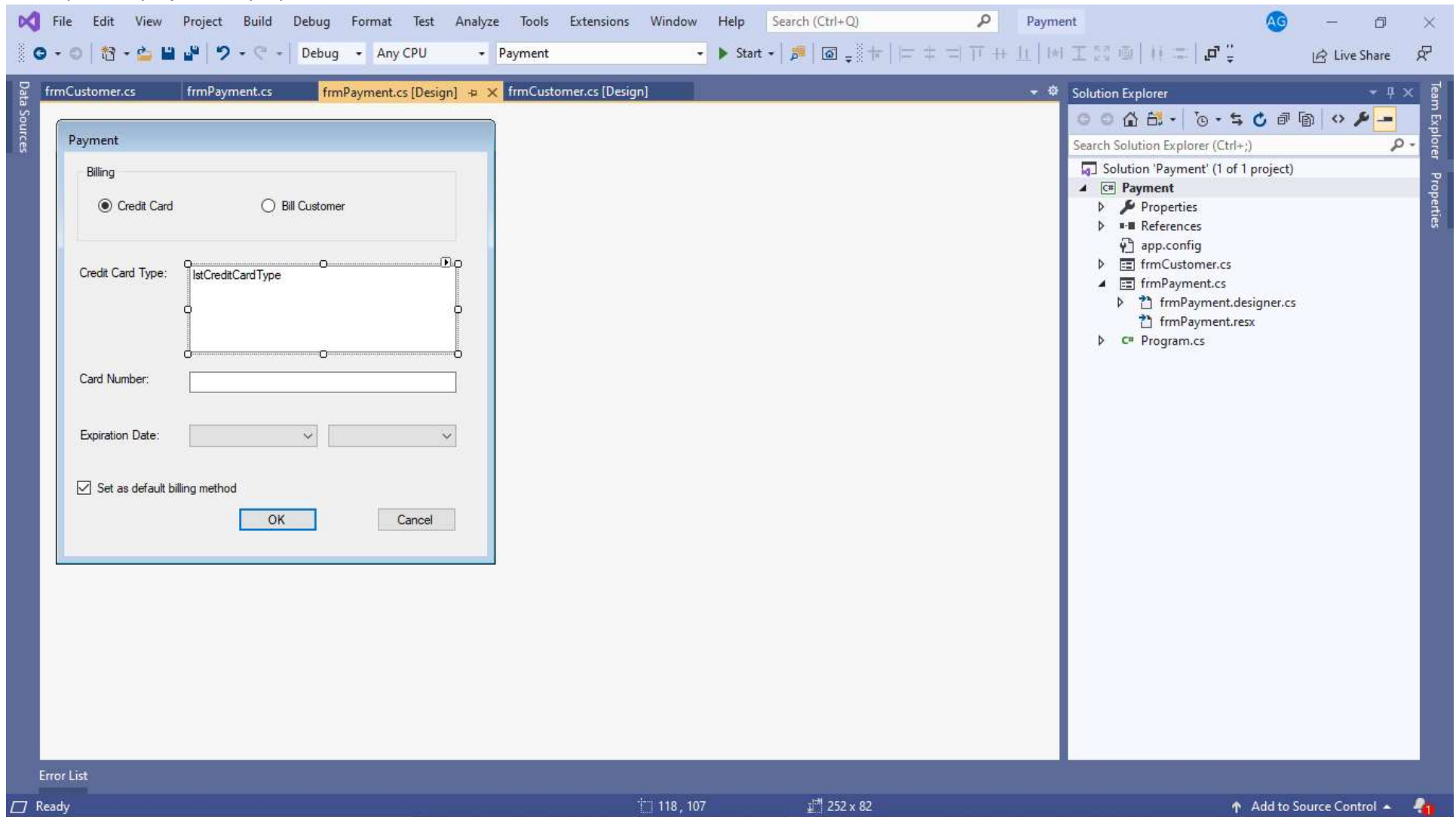
Monthly Investment must be in currency format.

OK

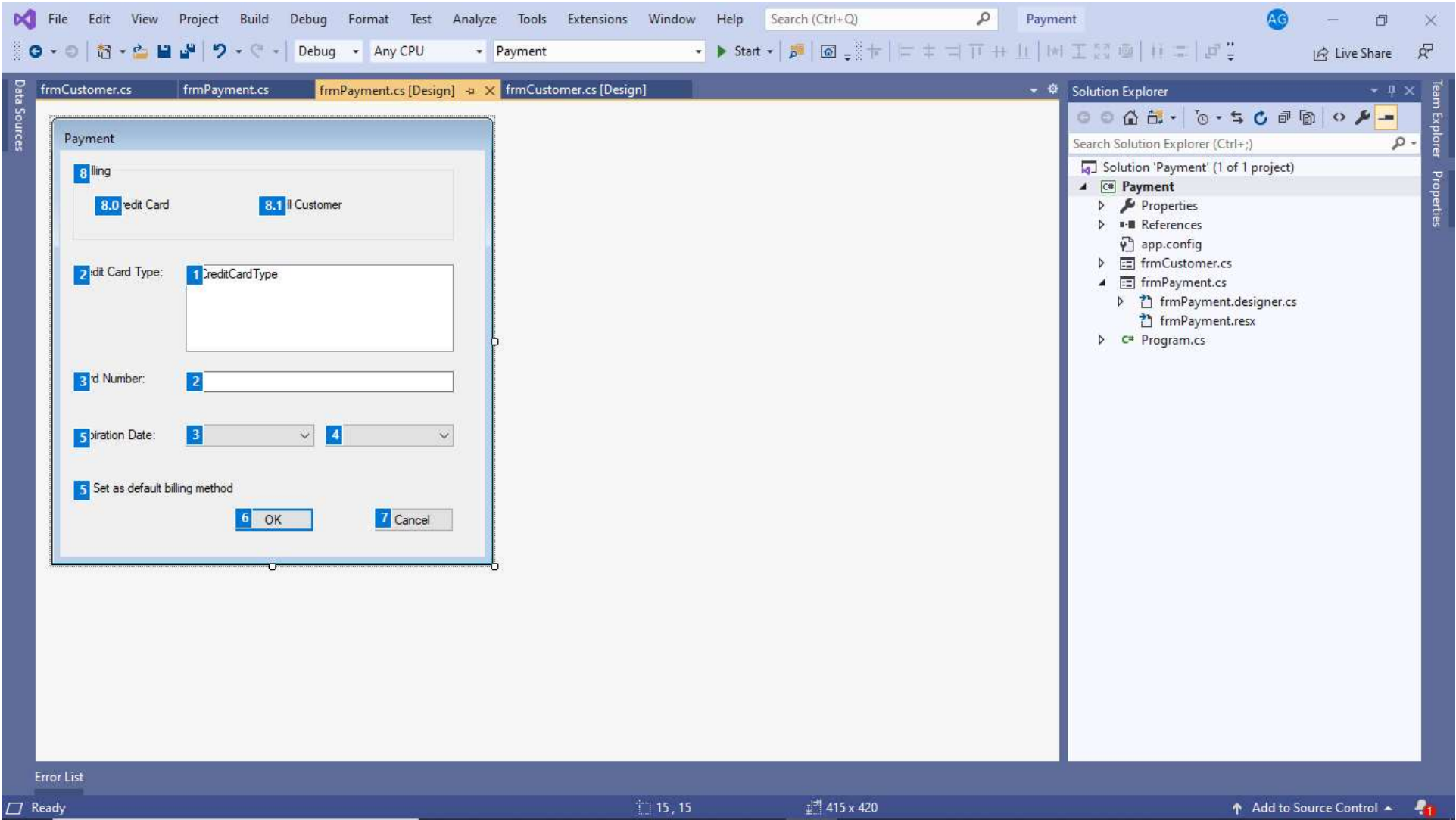
Exercise 10-1 Create the Payment application

Add code that provides for formatted enteries (#'s 2- 10)

#1-3 Open the project and prepare 2 forms



#4&5 Design the payment form



Add the code for the Customer form #6-9

The screenshot shows the Visual Studio IDE with the 'Payment' project selected. The code editor displays the 'frmCustomer.cs' file, which is a partial class for the 'frmCustomer' form. The code includes using statements for 'System.Threading.Tasks' and 'System.Windows.Forms', followed by a multi-line comment block. The 'namespace Payment' is defined, and the 'frmCustomer' class is declared as a partial class inheriting from 'Form'. The class contains three methods: a constructor 'frmCustomer()' that calls 'InitializeComponent()', a 'Load' event handler 'frmCustomer_Load' that adds 'Mike Smith' and 'Nancy Jones' to a 'cboNames' list, and a 'DataChanged' event handler that sets 'isDataSaved' to false. The 'btnSelectPayment_Click' method is also shown, starting with 'Form paymentForm = new frmPayment();'. The Solution Explorer on the right shows the project structure, including 'app.config', 'frmCustomer.cs', 'frmPayment.cs', 'frmPayment.designer.cs', 'frmPayment.resx', and 'Program.cs'. The status bar at the bottom indicates '105 %' zoom, 'No issues found', and 'Ln: 41 Ch: 55 SPC CRLF'.

```
9 using System.Threading.Tasks;
10 using System.Windows.Forms;
11 /*****
12  * Andrea Griffis
13  * Assignment:
14  * 10-1 Create the Payment Application
15  * *****/
16 namespace Payment
17 {
18     3 references
19     public partial class frmCustomer : Form
20     {
21         1 reference
22         public frmCustomer()
23         {
24             InitializeComponent();
25         }
26         bool isDataSaved = true;
27
28         1 reference
29         private void frmCustomer_Load(object sender, EventArgs e)
30         {
31             cboNames.Items.Add("Mike Smith");
32             cboNames.Items.Add("Nancy Jones");
33         }
34
35         2 references
36         private void DataChanged(object sender, EventArgs e)
37         {
38             isDataSaved = false;
39         }
40
41         1 reference
42         private void btnSelectPayment_Click(object sender, EventArgs e)
43         {
44             Form paymentForm = new frmPayment();
45         }
46     }
47 }
```

105 % No issues found Ln: 41 Ch: 55 SPC CRLF

Ready Add to Source Control

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Payment

Debug Any CPU Payment Start

frmCustomer.cs frmPayment.cs frmPayment.cs [Design] frmCustomer.cs [Design]

Payment Payment.frmCustomer DataChanged(object sender, EventArgs e)

```
35 1 reference
36 private void btnSelectPayment_Click(object sender, EventArgs e)
37 {
38     Form paymentForm = new frmPayment();
39     DialogResult selectedButton = paymentForm.ShowDialog();
40     if (selectedButton == DialogResult.OK)
41     {
42         lblPayment.Text = (string)paymentForm.Tag;
43     }
44 }
45 1 reference
46 private void btnSave_Click(object sender, EventArgs e)
47 {
48     if (IsValidData())
49     {
50         SaveData();
51     }
52 }
53 2 references
54 private void SaveData()
55 {
56     cboNames.SelectedIndex = -1;
57     lblPayment.Text = "";
58     isDataSaved = true;
59     cboNames.Focus();
60 }
61 2 references
62 private bool IsValidData()
63 {
64     if (cboNames.SelectedIndex == -1)
```

105 % No issues found Ln: 34 Ch: 10 SPC CRLF

Error List

Ready Add to Source Control

Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'Payment' (1 of 1 project)

- Payment
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs

Visual Studio interface showing the code for `frmPayment.cs` in the `Payment` project.

Code Editor:

```
61 private bool IsValidData()
62 {
63     if (cboNames.SelectedIndex == -1)
64     {
65         MessageBox.Show("You must select a customer.", "Entry Error");
66         cboNames.Focus();
67         return false;
68     }
69     if (lblPayment.Text == "")
70     {
71         MessageBox.Show("You must enter a payment.", "Entry Error");
72         return false;
73     }
74     return true;
75 }
76 private void btnExit_Click(object sender, EventArgs e)
77 {
78     this.Close();
79 }
80
81 private void frmCustomer_FormClosing(object sender,
82     FormClosingEventArgs e)
83 {
84     if (isDataSaved == false)
85     {
86         string message =
87             "This form contains unsaved data. \n\n" +
88             "Do you want to save it?";
89
90         DialogResult button =
```

Solution Explorer:

- Solution 'Payment' (1 of 1 project)
 - Payment
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs

Status Bar: 105 % | No issues found | Ln: 34 | Ch: 10 | SPC | CRLF

Visual Studio interface showing a C# code file named `frmPayment.cs` in the `Payment` project. The code implements the `frmCustomer_FormClosing` event handler, which checks if data is saved before closing the form. If data is not saved, a message box is shown to the user, and the form closing is canceled if the user chooses "Yes" or "Cancel".

```
79 }
80
81 0 references
82 private void frmCustomer_FormClosing(object sender,
83     FormClosingEventArgs e)
84 {
85     if (isDataSaved == false)
86     {
87         string message =
88             "This form contains unsaved data. \n\n" +
89             "Do you want to save it?";
90
91         DialogResult button =
92             MessageBox.Show(message, "Customer",
93                 MessageBoxButtons.YesNoCancel,
94                 MessageBoxIcon.Warning);
95
96         if (button == DialogResult.Yes)
97         {
98             if (IsValidData())
99                 this.SaveData();
100             else
101                 e.Cancel = true;
102         }
103         if (button == DialogResult.Cancel)
104         {
105             e.Cancel = true;
106         }
107     }
108 }
109 }
```

The Solution Explorer on the right shows the project structure for `Payment`, including `app.config`, `frmCustomer.cs`, `frmPayment.cs`, `frmPayment.designer.cs`, `frmPayment.resx`, and `Program.cs`.

At the bottom, the status bar indicates "105 %", "No issues found", and "Ln: 34 Ch: 10 SPC CRLF". The Error List is empty, and the status is "Ready".

Properties

cboNames System.Windows.Forms.ComboBox

Move	
ParentChanged	
PreviewKeyDown	
QueryAccessibilityHelp	
QueryContinueDrag	
RegionChanged	
Resize	
RightToLeftChanged	
SelectedIndexChanged	DataChanged
SelectedValueChanged	
SelectionChangeCommitted	
SizeChanged	

SelectedIndexChanged
Occurs when the value of the SelectedIndex property changes.

Properties

lblPayment System.Windows.Forms.Label

Resize	
RightToLeftChanged	
SizeChanged	
StyleChanged	
SystemColorsChanged	
TabIndexChanged	
TextAlignChanged	
TextChanged	DataChanged
Validated	
Validating	
VisibleChanged	

TextChanged
Event raised when the value of the Text property is changed on Control.

Test the application

Payment

Billing

☒ Credit Card ☐ Bill Customer

Credit Card Type:

Card Number:

Expiration Date:

☒ Set as default billing method

OK Cancel

Customer

Customer name:

Payment method:

Select Payment

Save Exit

Add code to the Payment form #10-13

The screenshot shows the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q)
- Toolbar:** Includes buttons for Run, Stop, Break, and other development actions.
- Tab Bar:** Shows open files: frmCustomer.cs, frmPayment.cs, frmPayment.cs [Design], and frmCustomer.cs [Design].
- Code Editor:** Displays the code for `frmPayment.cs`. The code is as follows:

```
9 using System.Threading.Tasks;
10 using System.Windows.Forms;
11
12 namespace Payment
13 {
14     3 references
15     public partial class frmPayment : Form
16     {
17         1 reference
18         public frmPayment()
19         {
20             InitializeComponent();
21         }
22         /*****
23          * Andrea Griffis
24          * Assignment:
25          * 10-1 Create the Payment Application
26          * *****/
27         1 reference
28         private void frmPayment_Load(object sender, EventArgs e)
29         {
30             lstCreditCardType.Items.Add("Visa");
31             lstCreditCardType.Items.Add("MasterCard");
32             lstCreditCardType.Items.Add("America Express");
33             lstCreditCardType.SelectedIndex = 0;
34
35             string[] months = {"Select a Month...",
36                               "January", "February", "March", "April", "May",
37                               "June", "July", "August", "September", "October",
38                               "November", "December"};
39             foreach (string month in months)
40                 cboExpirationMonth.Items.Add(month);
41             cboExpirationMonth.SelectedIndex = 0;
42         }
43     }
44 }
```
- Solution Explorer:** Shows the project structure for 'Payment' (1 of 1 project):
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs
- Team Explorer:** Shows the 'Properties' tab.
- Status Bar:** Displays '105 %', 'No issues found', and 'Ln: 11 Ch: 1 SPC CRLF'.
- Error List:** Shows 'Item(s) Saved'.
- Bottom Bar:** Includes 'Add to Source Control' and a notification icon.

Visual Studio interface showing the code editor for `frmPayment.cs` in the `Payment` project. The code defines a `frmPayment` class with a `Payment` constructor and two methods: `btnOk_Click` and `IsValidData`.

```
40 int year = DateTime.Today.Year;
41 int endYear = year + 8;
42 cboExpirationYear.Items.Add("Select a Year...");
43 while (year < endYear)
44 {
45     cboExpirationYear.Items.Add(year);
46     year++;
47 }
48 cboExpirationYear.SelectedIndex = 0;
49 }
50
51 1 reference
52 private void btnOk_Click(object sender, EventArgs e)
53 {
54     if (IsValidData())
55     {
56         this.SaveData();
57     }
58 }
59 1 reference
60 private bool IsValidData()
61 {
62     if (rdoCreditCard.Checked)
63     {
64         if (lstCreditCardType.SelectedIndex == -1)
65         {
66             MessageBox.Show("You must enter a credit card type.",
67                             "Entry Error");
68             lstCreditCardType.Focus();
69             return false;
70         }
71         if (txtCardNumber.Text == "")
```

The Solution Explorer on the right shows the project structure:

- Payment (1 of 1 project)
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs

The status bar at the bottom indicates 105% zoom, no issues found, and the file encoding is UTF-8 (SPC, CRLF).

Visual Studio interface showing the code for `frmPayment.cs` in the `Payment` project.

Code Snippet:

```
70 if (txtCardNumber.Text == "")
71 {
72     MessageBox.Show("You must enter a credit card number.",
73         "Entry Error");
74     txtCardNumber.Focus();
75     return false;
76 }
77 if (cboExpirationMonth.SelectedIndex == 0)
78 {
79     MessageBox.Show("You must select a Month.",
80         "Entry Error");
81     cboExpirationMonth.Focus();
82     return false;
83 }
84 if (cboExpirationYear.SelectedIndex == 0)
85 {
86     MessageBox.Show("You must select a year.",
87         "Entry Error");
88     cboExpirationYear.Focus();
89     return false;
90 }
91 }
92 return true;
93 }
94
95 1 reference
96 private void SaveData()
97 {
98     string msg = null;
99     if (rdoCreditCard.Checked == true)
100     {
101         msg += "Charge to credit card." + "\n\n";
102         msg += "Card type: " + lstCreditCardType.Text + "\n";
103     }
104 }
```

Solution Explorer:

- Payment
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs

Status Bar: 105 % | No issues found | Ln: 11 | Ch: 1 | SPC | CRLF

Visual Studio interface showing the code for `frmPayment.cs` in the `Payment` project.

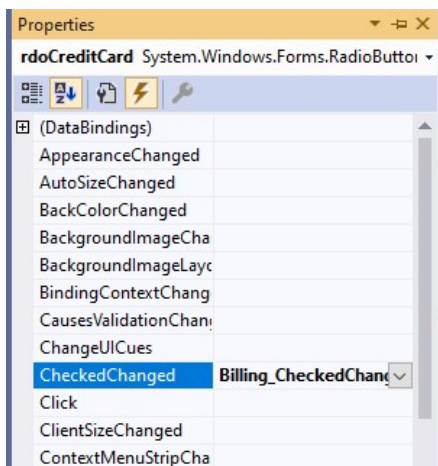
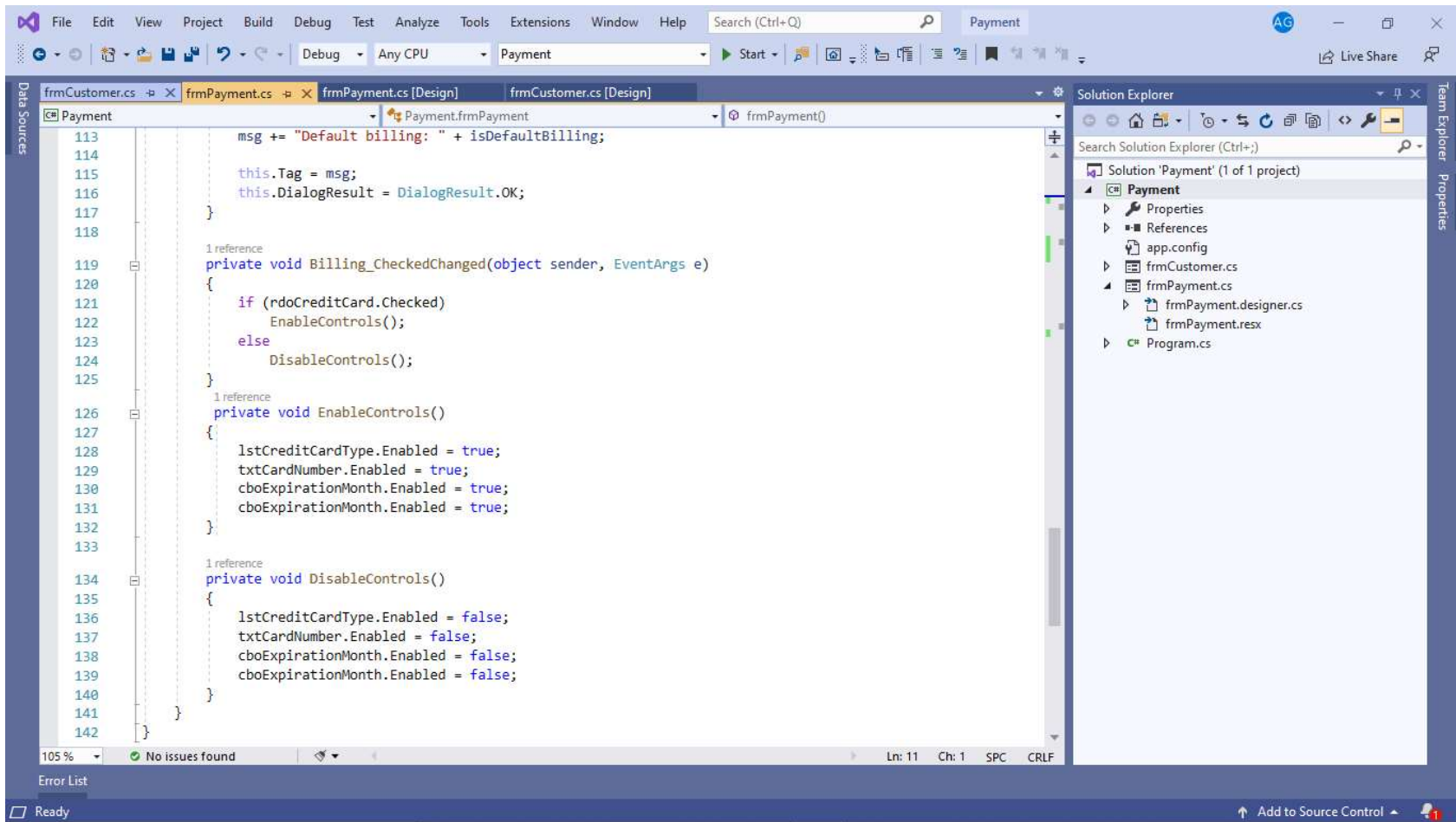
Code in `frmPayment.cs`:

```
97 string msg = null;
98 if (rdoCreditCard.Checked == true)
99 {
100     msg += "Charge to credit card." + "\n\n";
101     msg += "Card type: " + lstCreditCardType.Text + "\n";
102     msg += "Card number: " + txtCardNumber.Text + "\n";
103     msg += "Expiration date : "
104           + cboExpirationMonth.Text + "/"
105           + cboExpirationYear.Text + "\n";
106 }
107 else
108 {
109     msg += "Send bill to customer." + "\n";
110 }
111
112 bool isDefaultBilling = chkDefault.Checked;
113 msg += "Default billing: " + isDefaultBilling;
114
115 this.Tag = msg;
116 this.DialogResult = DialogResult.OK;
117 }
118
119 1 reference
120 private void Billing_CheckedChanged(object sender, EventArgs e)
121 {
122     if (rdoCreditCard.Checked)
123         EnableControls();
124     else
125         DisableControls();
126 }
127
128 1 reference
129 private void EnableControls()
```

Solution Explorer:

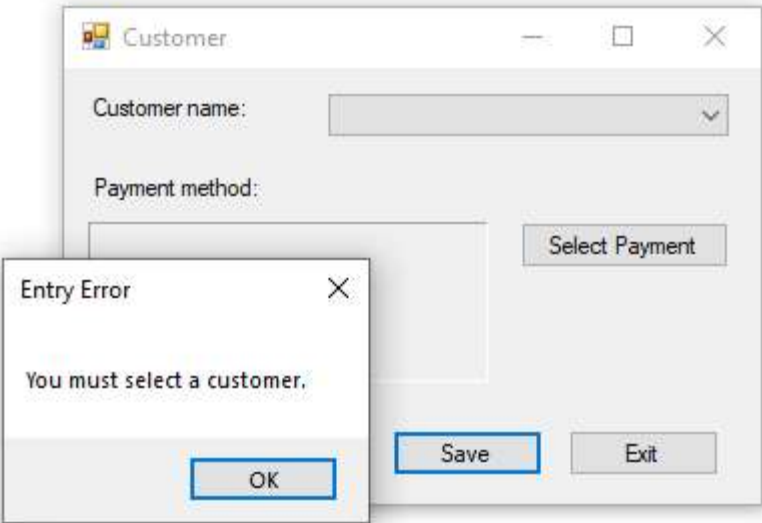
- Payment
 - Properties
 - References
 - app.config
 - frmCustomer.cs
 - frmPayment.cs
 - frmPayment.designer.cs
 - frmPayment.resx
 - Program.cs

Status Bar: 105 % | No issues found | Ln: 11 | Ch: 1 | SPC | CRLF

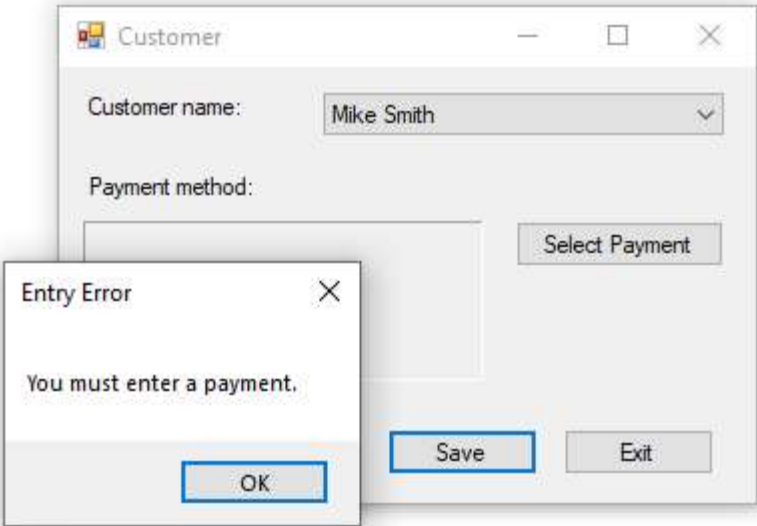


Test the Application

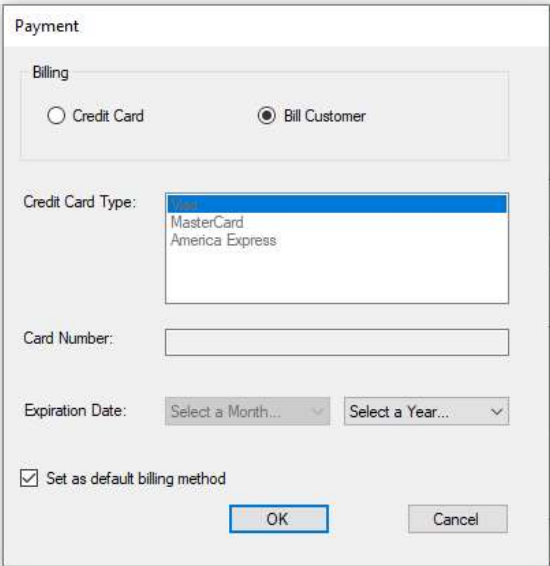
1) No Customer Selected 2) Cust Selected, No payment selected 3) Bill to customer selected 4)No Card # entered 5) No month selected



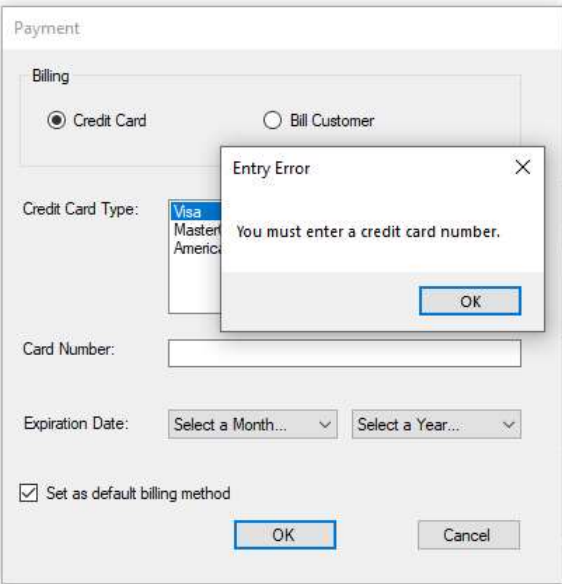
1.



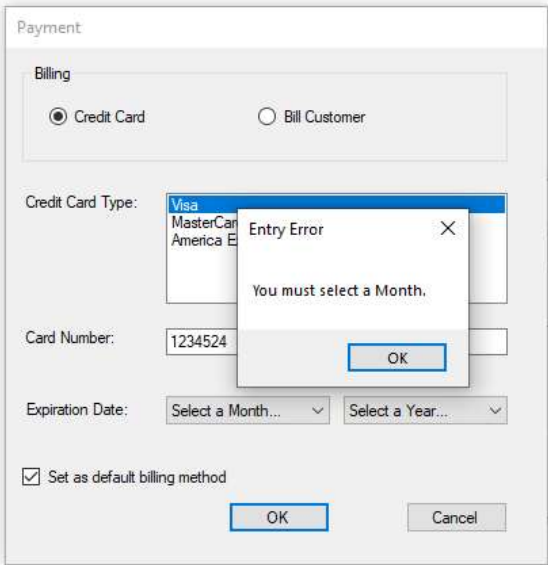
2.



3.



4.



5.

Test the application Cont.

6) No Year selected 7) Completed payment form, set as default checked 8) Completed payment form, set as default unchecked

6.

Payment

Billing

☒ Credit Card☐ Bill Customer

Credit Card Type:

Visa
MasterC
America

Card Number:

1233456

Expiration Date:

January

Select a Year...

☒ Set as default billing method

OK

Cancel

Entry Error

You must select a year.

OK

7.

Customer

Customer name:

Mike Smith

Payment method:

Charge to credit card.

Card type: Visa
Card number: 1233456
Expiration date : January/2022
Default billing: True

Select Payment

Save

Exit

8.

Customer

Customer name:

Mike Smith

Payment method:

Charge to credit card.

Card type: Visa
Card number: 1233456
Expiration date : January/2022
Default billing: False

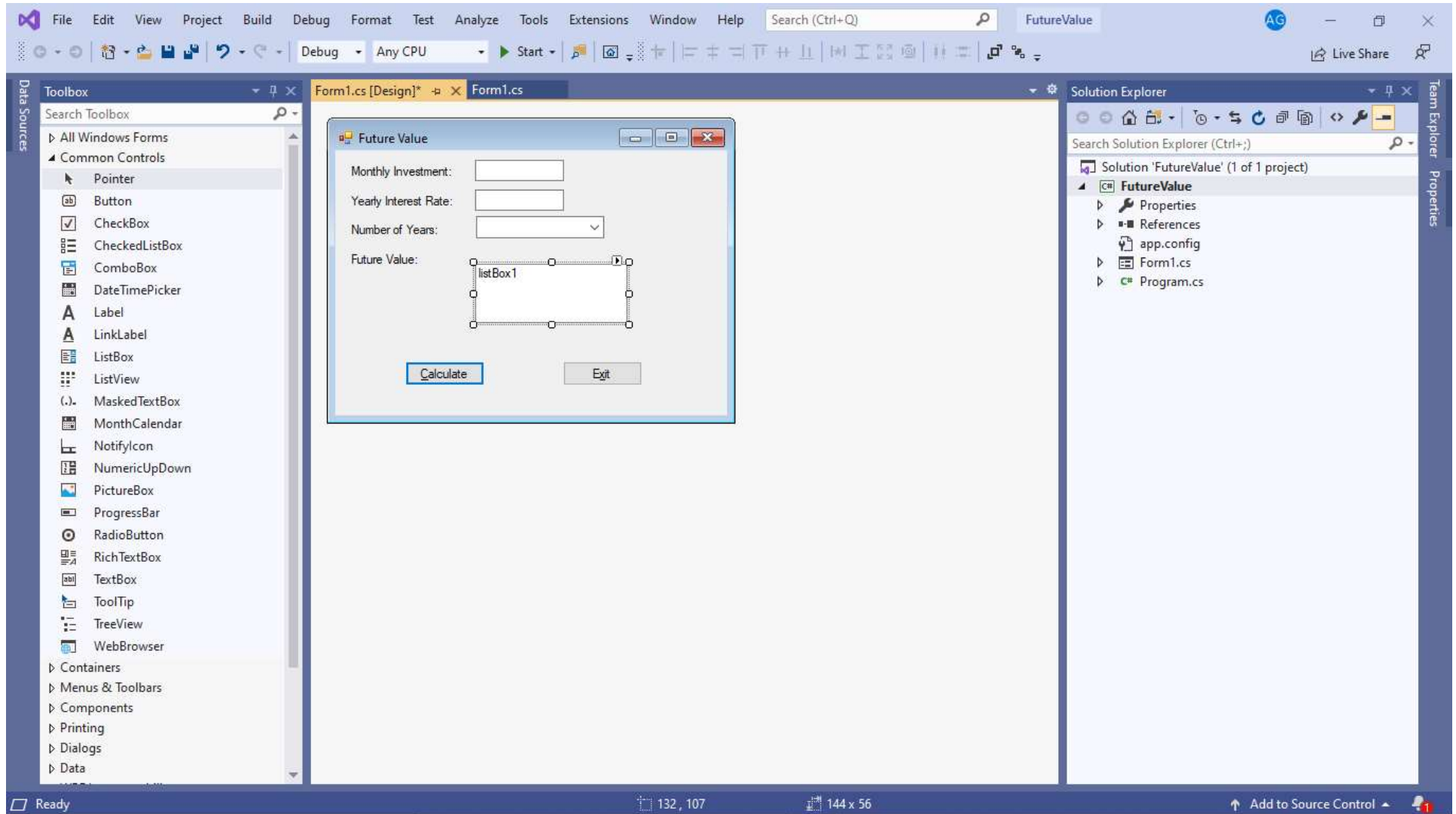
Select Payment

Save

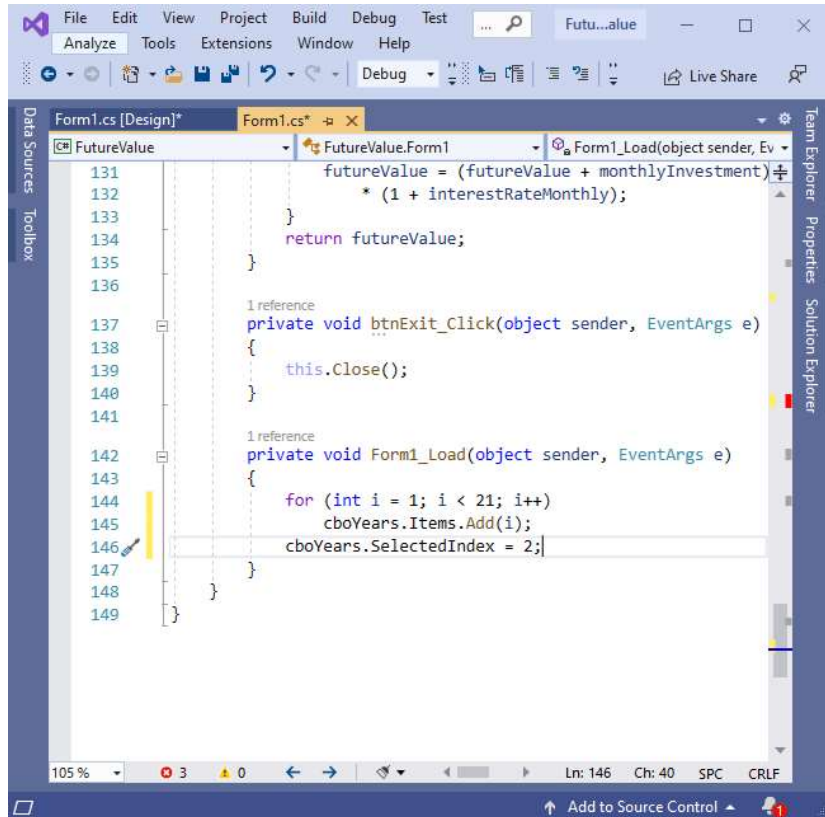
Exit

Exercise 10-2 Enhance the FV application

Open the future value

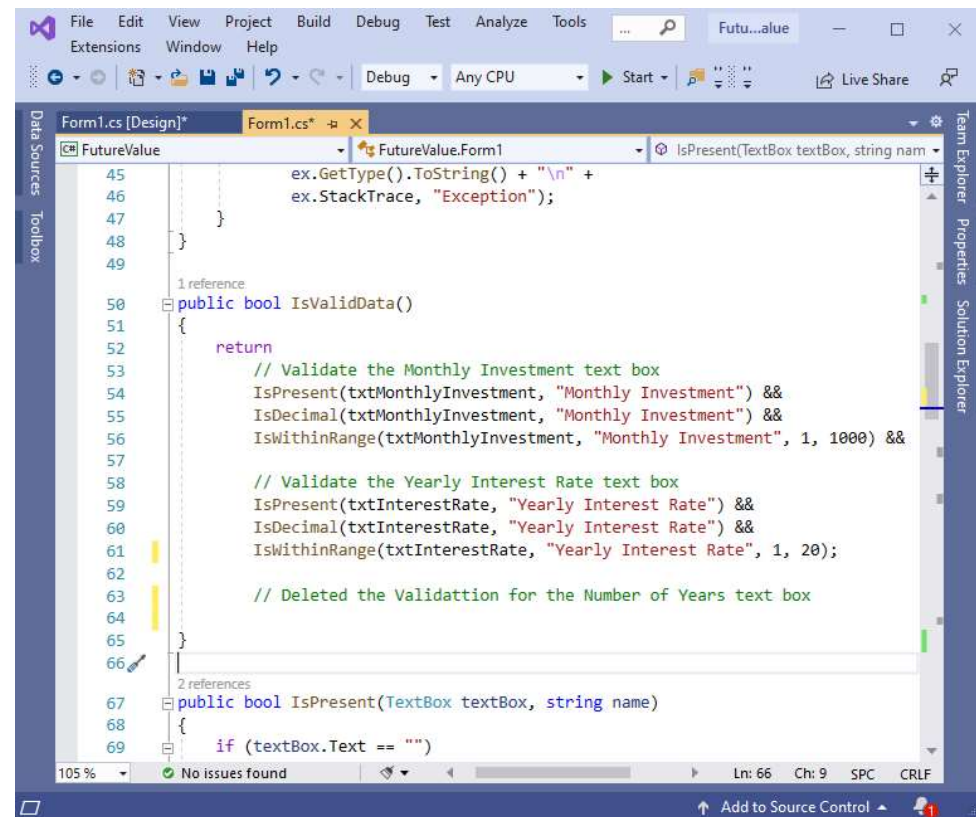


Add Code that works with the controls



```
File Edit View Project Build Debug Test ... Futu...alue
Analyze Tools Extensions Window Help
Debug Any CPU Start Live Share

Form1.cs [Design]* Form1.cs*
FutureValue FutureValue.Form1 Form1_Load(object sender, EventArgs e)
131 futureValue = (futureValue + monthlyInvestment)
132 * (1 + interestRateMonthly);
133 }
134 return futureValue;
135 }
136
137 1 reference
138 private void btnExit_Click(object sender, EventArgs e)
139 {
140     this.Close();
141 }
142
143 1 reference
144 private void Form1_Load(object sender, EventArgs e)
145 {
146     for (int i = 1; i < 21; i++)
147         cboYears.Items.Add(i);
148     cboYears.SelectedIndex = 2;
149 }
```



```
File Edit View Project Build Debug Test Analyze Tools ... Futu...alue
Extensions Window Help
Debug Any CPU Start Live Share

Form1.cs [Design]* Form1.cs*
FutureValue FutureValue.Form1 IsPresent(TextBox textBox, string name)
45 ex.GetType().ToString() + "\n" +
46 ex.StackTrace, "Exception");
47 }
48
49
50 1 reference
51 public bool IsValidData()
52 {
53     return
54         // Validate the Monthly Investment text box
55         IsPresent(txtMonthlyInvestment, "Monthly Investment") &&
56         IsDecimal(txtMonthlyInvestment, "Monthly Investment") &&
57         IsWithinRange(txtMonthlyInvestment, "Monthly Investment", 1, 1000) &&
58         // Validate the Yearly Interest Rate text box
59         IsPresent(txtInterestRate, "Yearly Interest Rate") &&
60         IsDecimal(txtInterestRate, "Yearly Interest Rate") &&
61         IsWithinRange(txtInterestRate, "Yearly Interest Rate", 1, 20);
62         // Deleted the Validation for the Number of Years text box
63 }
64
65
66 2 references
67 public bool IsPresent(TextBox textBox, string name)
68 {
69     if (textBox.Text == "")
```

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Futu...alue

Debug Any CPU Start

Form1.cs [Design]* Form1.cs* x

FutureValue FutureValue.Form1 btnCalculate_Click(object sender, EventArgs e)

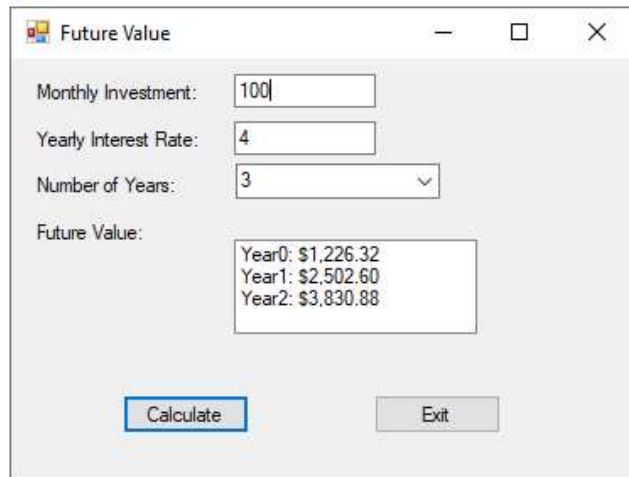
```
19
20 1 reference
21 private void btnCalculate_Click(object sender, EventArgs e)
22 {
23     try
24     {
25         if (IsValidData())
26         {
27             decimal monthlyInvestment =
28                 Convert.ToDecimal(txtMonthlyInvestment.Text);
29             decimal yearlyInterestRate =
30                 Convert.ToDecimal(txtInterestRate.Text);
31             int years =
32                 Convert.ToInt32(cboYears.Text);
33
34             int months = years * 12;
35             decimal monthlyInterestRate = yearlyInterestRate / 12 / 100;
36             // updated to clear the list and recieve # of yrs from combo box and future value for each year
37             lstFutureValues.Items.Clear();
38             decimal futureValue = 0m;
39             for (int i = 0; i < months; i++)
40             {
41                 futureValue = (futureValue + monthlyInvestment)
42                     * (1 + monthlyInterestRate);
43                 if ((i+1) % 12 == 0)
44                 {
45                     int year = (i - 1) / 12;
46                     lstFutureValues.Items.Add("Year" + year + ": "
47                         + futureValue.ToString("c"));
48                 }
49             }
50             txtMonthlyInvestment.Focus();
51         }
52     }
53 }
```

105 % No issues found Ln: 32 Ch: 21 SPC CRLF

Ready Add to Source Control

Test the application

1) Completed form correctly 2) showing 3 as default with whole list 3) no month chosen 4) interest rate invalid 5) completed form w/scroll on list box



Future Value

Monthly Investment: 100

Yearly Interest Rate: 4

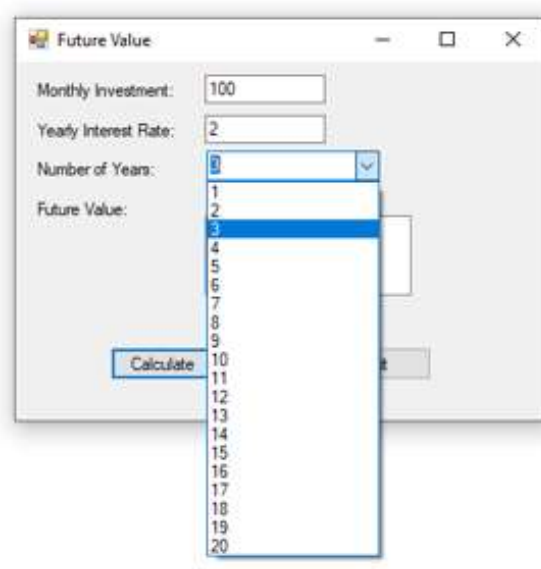
Number of Years: 3

Future Value:

Year0: \$1,226.32
Year1: \$2,502.60
Year2: \$3,830.88

Calculate Exit

1.



Future Value

Monthly Investment: 100

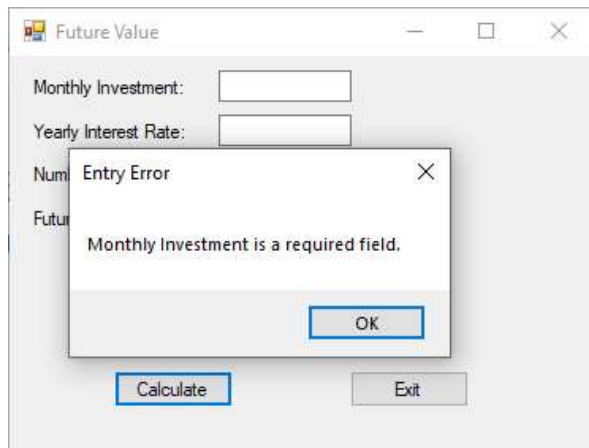
Yearly Interest Rate: 2

Number of Years: 3

Future Value:

Calculate

2.



Future Value

Monthly Investment:

Yearly Interest Rate:

Number of Years:

Future Value:

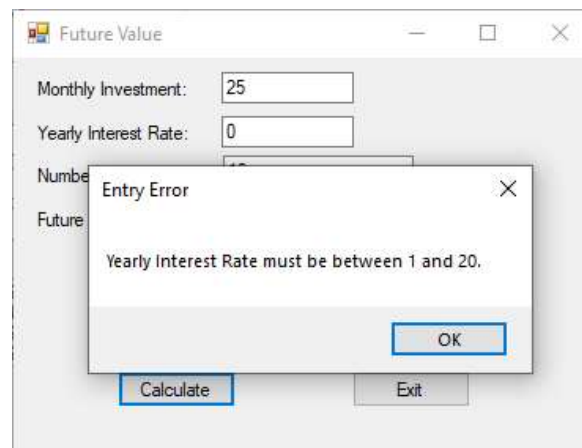
Entry Error

Monthly Investment is a required field.

OK

Calculate Exit

3.



Future Value

Monthly Investment: 25

Yearly Interest Rate: 0

Number of Years:

Future Value:

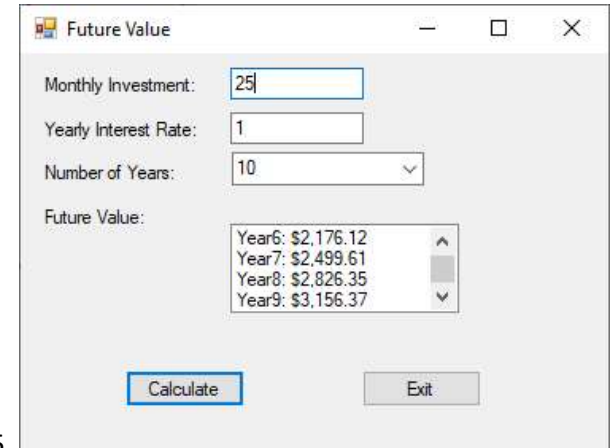
Entry Error

Yearly Interest Rate must be between 1 and 20.

OK

Calculate Exit

4.



Future Value

Monthly Investment: 25

Yearly Interest Rate: 1

Number of Years: 10

Future Value:

Year6: \$2,176.12
Year7: \$2,499.61
Year8: \$2,826.35
Year9: \$3,156.37

Calculate Exit

5.